Contextual Pre-planning on Reward Machine Abstractions for Enhanced Transfer in Deep Reinforcement Learning

Guy Azran[1], **Mohamad H. Danesh**[2], **Stefano V. Albrecht**[3], **Sarah Keren**[1]

[1]Taub Faculty of Computer Science, Technion – Israel Institute of Technology
[2]School of Computer Science, McGill University
[3]School of Informatics, University of Edinburgh
guy.azran@campus.technion.ac.il, mo.danesh@mail.mcgill.ca, s.albrecht@ed.ac.uk, sarahk@cs.technion.ac.il

Abstract

Recent studies show that deep reinforcement learning (DRL) agents tend to overfit to the task on which they were trained and fail to adapt to minor environment changes. To expedite learning when transferring to unseen tasks, we propose a novel approach to representing the current task using *reward machines* (RMs), state machine abstractions that induce subtasks based on the current task's rewards and dynamics. Our method provides agents with symbolic representations of optimal transitions from their current abstract state and rewards them for achieving these transitions. These representations are shared across tasks, allowing agents to exploit knowledge of previously encountered symbols and transitions, thus enhancing transfer. Empirical results show that our representations improve sample efficiency and few-shot transfer in a variety of domains.

1 Introduction

Reinforcement learning (RL) methods, especially deep RL (DRL) methods, have shown impressive capabilities in a wide variety of problems (Chen, Xu, and Agrawal 2021; Schrittwieser et al. 2020). However, recent studies show that these algorithms have difficulty adapting to even the slightest variations in the agent's objective or environment dynamics (Danesh and Fern 2022; Agarwal et al. 2020; Zhang et al. 2018; Leike et al. 2017). Adapting quickly to new tasks is imperative in real-world scenarios, such as robotics (Dunion et al. 2023a,b) and healthcare (Tseng et al. 2017), where agents reside in a dynamic world with ever-changing objectives and constraints. Consequently, agents require many interactions with the environment to learn to perform new tasks despite having mastered similar ones. The problem is exacerbated for tasks with sparse reward signals (Gupta et al. 2022) and long-term dependencies between actions (Langford 2018).

Example 1 *A housekeeper robot learns to do multiple tasks, one of which is to make coffee in a mug. Next, the robot is tasked with making coffee in a glass, something it has never attempted. The two tasks are similar in that they interact with many of the same objects (e.g. coffee, spoon, etc.) and perform identical subtasks (e.g. boil water, fill cup, etc.). The robot is expected to use its experience in making coffee in a mug to learn to achieve the new task more quickly.*

A *contextual MDP* (CMDP) (Langford 2017; Hallak, Di Castro, and Mannor 2015) models settings like Example 1 as a collection of tasks in the same environment, where each task is represented by the current *context*. CMDPs have been used in recent work that aims to improve *zero-shot* transfer capabilities, i.e., solving new tasks after training on a subset of them (Benjamins et al. 2022; Hallak, Di Castro, and Mannor 2015). In contrast, we aim to improve *few-shot* transfer, in which the agent may continue training on previously unseen tasks with the objective of minimizing the additional training required to achieve desirable performance.

One of the key challenges when using a CMDP to model transfer learning settings is finding a concise way to represent the current context while maximizing transfer capabilities. For this, we take advantage of *reward machines* (RMs) (Toro Icarte et al. 2018), state-machine-based abstractions that represent the structure of the reward function and the dynamics of a task and its subtasks. Transitions between abstract states in the RM occur when certain facts, represented as binary symbols, hold true. As the agent traverses the environment, it keeps track of these facts and its current RM state. Camacho et al. (2021) used RMs by providing the agent with the current abstract state and showed that this can expedite learning on a single task. In contrast, we leverage RMs to improve transfer to new tasks.

Our novel technique, called *Contextual PRE-Planning* (C-PREP), takes as input a CMDP and an RM generator function that represents contextual information through task-specific RM abstractions with shared symbolic representations. Given a task, C-PREP finds an optimal policy in the corresponding RM abstraction and gives the agent the next desired abstract transition according to that policy as additional input. Furthermore, C-PREP uses the RM by reshaping the reward function according to abstract state transitions within the RM, thus highlighting important transitions throughout learning. When transferred to a new task, the agent can exploit abstract transitions that it has encountered during training and needs only to adapt to symbols with which it has not previously interacted.

We empirically evaluate C-PREP in various environments with sparse rewards and varying difficulties. In our experiments, a DQN agent (Mnih et al. 2015) is initially trained on a collection of source contexts. Subsequently, we transfer the policy network to a different set of target contexts, where it undergoes further training and evaluation. We observe an improvement in few-shot as well as zero-shot transfer performance when using C-PREP compared to other context representation methods. The performance gap grows as the problem difficulty increases, with improvements of 22.84% to 42.31% in time-to-threshold (few-shot transfer), and from 11.86% to 36.5% in jumpstart (zero-shot transfer) for the most complex tasks compared to the next best baseline.

2 Background

Reinforcement learning (RL) is a method for agent learning through experiencing the world, acting within it, and receiving rewards (both positive and negative) for achieving certain states or state transitions. RL problems commonly model the world as a **Markov decision process** (MDP) (Bellman 1957) $M = \langle S, A, T, R, \gamma \rangle$ where S is a set of possible states, A is a set of agent actions, $T : S \times A \times S \to [0, 1]$ is the state transition function, $R : S \times A \times S \to \mathbb{R}$ is the reward function, and γ is the temporal reward discount factor. The objective is to find a policy π^* such that: $\pi^* \in \arg\max_\pi \mathbb{E}[J(\pi)]$, where $J(\pi) = \mathbb{E}_{s_t, s_{t+1} \sim T; a_t \sim \pi} [\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1})]$ is the expected return of policy π.

In this work, we focus on **transfer learning** (TL), which is the improvement of learning a new task through the transfer of knowledge from a related task that has already been learned (Torrey and Shavlik 2009). We model a collection of MDPs using a **contextual MDP** (CMDP) (Hallak, Di Castro, and Mannor 2015), a 4-tuple $\langle C, S, A, \mathcal{M} \rangle$ where C is the context space, S and A are state and action spaces, and $\mathcal{M}$ is a mapping from a context $c \in C$ to an MDP $\mathcal{M}_c$ consisting of S and A but with distinct transition and reward functions, that is, $\mathcal{M}_c = \langle S, A, T_c, R_c, \gamma \rangle$. We sometimes refer to context-induced MDPs as "tasks", and to the shared S and A as the "environment". In Example 1, C is the set of all house chores, S and A are the state of the house and the agent's capabilities, and $\mathcal{M}$ maps a chore c to an MDP $\mathcal{M}_c$ that corresponds to completing the chore.

Fig. 1a depicts the general flow of transfer learning over a CMDP. The input is the observed MDP state and the current context. Optionally, the state representation is processed by a feature extractor to be represented as a vector. The context is represented via a *context representation function* that maps a context to a vector representation. The state representation is merged with the context representation (usually by concatenation), and the new representation is fed into a policy network that will determine the next action.

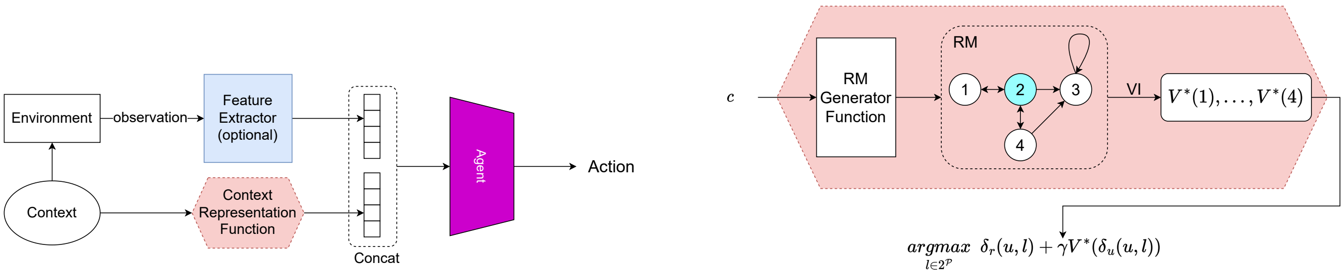
Kirk et al. (2021) distinguish between two categories of context representations. The first type, known as *controllable context representations* (CTL), includes the necessary information to generate the MDP, which can be thought of as a transparent implementation of the environment generation process (implemented in $\mathcal{M}$). The second type, *procedural content generation context representation* (PCG), conceals the MDP variables and only reveals information about the context identity, operating as a black box with no insight into the generation process.

Given a CMDP $\langle C, S, A, \mathcal{M} \rangle$, transfer learning algorithms attempt to leverage knowledge from interactions with a set of *source contexts* $C_{\text{src}} \subset C$ to improve learning in a set of *target contexts* $C_{\text{tgt}} \subset C$ such that $C_{\text{src}} \cap C_{\text{tgt}} = \emptyset$. In Example 1, C_{src} is the set of contexts representing the chores it learns to do, including making coffee in a mug. Making coffee in a glass is a context in C_{tgt}. Policies learned after training in C_{src} and C_{tgt} from scratch are the *source policy* and the *target policy*, respectively. The policy learned on C_{tgt} after training in C_{src} is the *transferred policy*. Given a distribution Ψ over C, the objective is to optimize a chosen *transfer utility* $\mathcal{U}$ in expectation over sampled source and target context sets. Transfer utilities of interest in this work, suggested by Taylor and Stone (2009), are *jumpstart* (JS), *time to threshold* (TT), and *transfer ratio* (TR). JS measures (zero-shot transfer) performance on the target contexts without additional training. TT measures the number of training steps taken until convergence to a policy of acceptable performance threshold (few-shot transfer). TR measures the ratio of rewards accumulated over time by the agent using knowledge transfer against the agent that is trained from scratch, that is, how much the agent benefits from transfer (transfer relevance). Precise calculations are in Appendix F.

Reward machines (RMs) (Toro Icarte et al. 2018) are state machine abstractions of MDPs. Given a set of propositional symbols $\mathcal{P}$, an RM is a 3-tuple $\mathfrak{R} = \langle U, \delta_u, \delta_r \rangle$ where U is a set of abstract states, and $\delta_u : U \times 2^{\mathcal{P}} \to U$ and $\delta_r : U \times 2^{\mathcal{P}} \to \mathbb{R}$ are the abstract transition and reward functions, respectively. Given the current abstract state $u \in U$ and a subset of propositional symbols $l \subseteq \mathcal{P}$ that hold true, $\delta_u(u, l)$ is the next abstract state and $\delta_r(u, l)$ is the reward received for this transition. When $\delta_u(u, l) = u'$, l is called the abstract *transition label* from u to u'. To connect between the abstraction and the underlying MDP, $\mathcal{P}$ is coupled with a *transition labeling function* $L : S \times A \times S \to 2^{\mathcal{P}}$ that maps state-action-state transitions in the MDP to abstract transition labels in the RM.

Fig. 2a textually describes an RM for the task of making coffee in Example 1. It defines abstract states u_0 to u_3 that each represents a high-level stage within the task of making a cup of coffee. The RM dictates that the agent must first boil some water, then put the coffee in the cup, and finally pour boiling water into the cup. These relationships are graphically visualized in Fig. 2b.

The main benefits of RMs are that they represent transitions between abstract states using binary symbols that pertain to the state of the underlying MDP (through L) and provide dense rewards via reward shaping. As a result, an RM corresponding to some context divides its induced task into sub-tasks that each describe a stage in the process of solving the overall task, rewarding the agent upon completion of each sub-task. To employ sensible reward shaping, we use *potential-based reward shaping* (Ng, Harada, and Russell 1999) which, given a potential function ϕ, defines a new abstract reward function $\delta'_r(u, l) = \delta_r(u, l) + \gamma\phi(\delta_u(u, l)) - \phi(u)$. Toro Icarte et al. (2022) prove that potential-based reward shaping guarantees that optimal policies in an MDP for which rewards have been replaced with RM rewards are optimal using the RM reshaped rewards. Moreover, it is em-



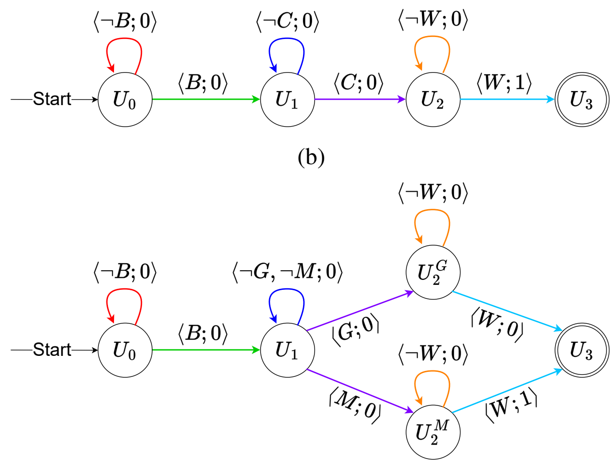
(a) (b)

Figure 1: (a) The general flow of transfer learning with a CMDP. (b) A visualization of the C-PREP context representation function. Context c is used to generate a task-specific RM.

```
Symbols:
    B - There is boiling water in the kettle.
    C - Coffee contents are in the cup.
    W - Boiling water poured into the cup.

Make-Coffee:
States - u0,u1,u2,u3
Transitions -
    (u0, not B) --> next=u0;r=0
    (u0,     B) --> next=u1;r=0
    (u1, not C) --> next=u1;r=0
    (u1,     C) --> next=u2;r=0
    (u2, not W) --> next=u2;r=0
    (u2,     W) --> next=u3;r=1
```

(a)



(c)

Figure 2: (a) A textual representation of the RM in Example 1 describing the `Make-Coffee` task. (b) A graph visualization of the textually defined RM. (c) An expansion of the RM that differentiates between mug and glass receptacles, described in Section 3.

pirically shown that using RM reshaped rewards can significantly expedite policy convergence for RL agents.

3 Contextual Pre-Planning (C-PREP) for Transfer Learning

We aim to improve transfer in multi-task domains modeled as CMDPs. Benjamins et al. (2022) proved that the policy must be conditioned on the context itself to guarantee optimality. Therefore, it is crucial to represent the context such that the agent can generalize across contexts. For this, we use RMs to represent contexts and offer a novel way to enhance the agent's ability to exploit its previous experiences in new settings. Since our focus is on exploiting the structure of the RMs for transfer and not on their generation, we assume that the RM generator function is given as input, which can be based on domain knowledge, learned from demonstration (Camacho et al. 2021), or learned via discrete optimization (Toro Icarte et al. 2019).

Camacho et al. (2021) exploited RMs to expedite learning in single-task domains by providing the agent with the current abstract state. We instead focus on transfer learning and provide the next desired abstract transition from the current RM abstract state as contextual input at each timestep. Essentially, we guide the agent through optimal paths in the RM with abstract transitions, represented using a set of symbols that is shared across all tasks. Upon transfer, the agent can expedite transfer learning by exploiting abstract transitions and leveraging prior knowledge of encountered symbols in the new task. This may be beneficial for learning in general but is key in transfer settings as it provides reusable representations between tasks.

C-PREP Context Representation Function. Based on the above intuition, we propose ***Contextual PRE-Planning*** (C-PREP) for leveraging information in context-specific RMs[1]. For each task, C-PREP generates an RM $\langle U, \delta_u, \delta_r\rangle$ with abstract transitions represented using a shared symbol set, i.e., all RM are defined using the symbol set $\mathcal{P}$. Using *value iteration* (VI) (Bellman 1957), we find an optimal policy in the RM as if it were a deterministic MDP. We note that VI is one of many possible choices, but we use it here to

[1]Code: https://github.com/CLAIR-LAB-TECHNION/C-PREP

compare to previous methods. We then give the agent an optimal abstract transition label in the RM from the current abstract state u (as dictated by the RM policy), i.e., a transition label l such that $\delta_u(u,l)$ is the next state on a (discounted) reward-maximizing path in the RM. Intuitively, we wish to guide the agent towards an optimal path within the RM.

C-PREP relies on providing the next desired abstract transition in the RM to the agent. However, since there is no direct representation of actions in the RM, standard VI does not apply. We, therefore, use a variant of VI, as suggested by Toro Icarte et al. (2022), with the following update rule over the abstract states of RM $\mathfrak{R}$.

$$V^k_{\mathfrak{R}}(u) = \max_{l\in 2^{\mathcal{P}}} \left[\delta_r(u,l) + \gamma V^{k-1}_{\mathfrak{R}}(\delta_u(u,l))\right] \quad (1)$$

where $V^k_{\mathfrak{R}}$ is the value of abstract state u at iteration k ($V^0_{\mathfrak{R}} = 0$), and $\delta_u(u,l)$ and $\delta_r(u,l)$ are the next abstract state and reward received for achieving transition label l at abstract state u, respectively. To show the relationship between this rule and VI for MDPs, we define $M_{\mathfrak{R}} = \langle U, 2^{\mathcal{P}}, T, R, \gamma\rangle$ where $T(u,l,\delta_u(u,l)) = 1$ and $R(u,l,u') = \delta_r(u,l)$. We observe that the VI update rule for $M_{\mathfrak{R}}$, denoted V^k, is equivalent to $V^k_{\mathfrak{R}}$. Formally,

$$\begin{aligned} V^k(u) &= \max_{l\in 2^{\mathcal{P}}} \sum_{u'\in U} T(u,l,u')(R(u,l,u') + \gamma V^{k-1}(u')) \\ &= \max_{l\in 2^{\mathcal{P}}} R(u,l,\delta_u(u,l)) + \gamma V^{k-1}(\delta_u(u,l)) \\ &= \max_{l\in 2^{\mathcal{P}}} \delta_r(u,l) + \gamma V^{k-1}(\delta_u(u,l))) = V^k_{\mathfrak{R}}(u) \end{aligned}$$

Thus, to identify optimal abstract transitions, we can find an abstract optimal policy in RM $\mathfrak{R}$ by using VI to find an optimal policy π^* in $M_{\mathfrak{R}}$.

Given the current abstract state u, from which there may be multiple optimal abstract transitions, C-PREP samples an optimal abstract transition l from $\pi^*(\cdot|u)$. Since π^* is optimal in deterministic MDP $M_{\mathfrak{R}}$:

$$supp(\pi^*(\cdot|u)) \subset \arg\max_{l\in 2^{\mathcal{P}}} (\delta_r(u,l) + \gamma V^*(\delta_u(u,l)))$$

where $supp(\pi^*(\cdot|u))$ is the support set of probability distribution $\pi^*(\cdot|u)$. Thus, any transition we sample from π^* is one that maximizes discounted return in the RM.

Based on the above formulations, the C-PREP context representation function (depicted in Fig. 1b) operates in a three-step process: (1) generate an RM $\mathfrak{R} = G(\mathcal{M}_c)$ for the current context c, (2) find an optimal policy π^* in $M_{\mathfrak{R}}$, (3) at each timestep, sample an optimal transition $l \sim \pi^*(\cdot, u)$ given the current RM abstract state u and return it.

Throughout training, the C-PREP RM generation function updates its returned representation according to the current abstract state. To notify the agent that a correct (or incorrect) abstract transition has been completed, we provide additional rewards that emphasize the executed abstract transition's quality. For this, we employ potential-based reward-shaping as defined in Section 2. As it is already calculated, we use V^* as the potential function ϕ to generate the reward signal that is provided to the agent instead of the original MDP reward. In the RM described in Fig. 2a, the agent will receive a higher reward for transitioning from state u_0 to u_1 rather than loop back to itself because this brings it closer to the abstract goal state.

Transfer Learning with C-PREP The input to our setting includes a CMDP $\langle C, S, A, \mathcal{M}\rangle$ and an *RM generator function* G that maps each context-induced task $\mathcal{M}_c$ to its corresponding RM $G(\mathcal{M}_c) = \langle U^c, \delta^c_u, \delta^c_r\rangle$ which is defined over shared symbol set $\mathcal{P}$.

The C-PREP context representation function can be integrated into any algorithm following the transfer learning flow depicted in Fig. 1a. Algorithm 1 (Appendix G) demonstrates an implementation of a DQN (Mnih et al. 2015) for transfer learning settings using C-PREP as the context representation function and RM reward shaping. The key differences between this implementation and the standard DQN are that the algorithm initially generates an RM for the sampled context, calculates its state values, and reshapes the RM rewards. States encountered in the episode are augmented by the C-PREP context representation according to the RM transition. Rewards are replaced with the reshaped rewards from the RM according to the achieved abstract transition at that timestep.

We note that the ability of C-PREP to support transfer depends on the *resolution* of the generated RMs, i.e., how well the generated RMs represent the context space. If the set of propositional symbols $\mathcal{P}$ is too abstract, the generated RMs do not sufficiently distinguish between contexts. In contrast, if it is too refined, computation time may increase due to running VI in huge tables for every context.

In Example 1, when training to make coffee in a mug, the agent learns to pour water into the mug and should exploit this capability upon transferring to the task of making coffee in a glass. Fig. 2 shows two different RMs that can be used to describe this setting. The RM in Fig. 2b does not differentiate between a mug and a glass, as they are both encapsulated by the "cup" symbol `C`. In contrast, the RM in Fig. 2c distinguishes between the tasks of making coffee in a mug and in a glass, rewarding the agent only for the former (when transitioning from u^M_2 to u_3). Including both mug and glass events demonstrates a case with two possible ways to perform a task, differentiated by the reward function.

4 Empirical Evaluation

The objective of our empirical evaluation is to examine whether agents using C-PREP exhibit improved performance on transfer utilities of interest.

Experimental Setup

Environments: We test our method in four environments with compound and long-horizon tasks and sparse reward signals[2]:

Grid Navigation (GN): An agent must reach a specified destination on a grid. The state space consists of the agent's current location and the action space includes moving in one of the four cardinal directions and a "done" action to be called upon arrival at the destination.

[2]Our code base is described in Appendix I.

Multi Points-of-Interest (MP): The agent navigates to multiple destinations *in any order*. The state space consists of the agent's location and an indicator of whether a certain destination has already been visited. The action space is as in GN but with an "arrived" action replacing "done".

Pick-Up and Drop-Off (PD): An agent picks up and drops off passengers at their destinations. The state space is as in MP with indicators for passengers that have been dropped off at their destinations. In addition to navigation actions, the action space contains "pick-up" and "drop-off" actions.

Ordered Navigation (ON): The agent must navigate to specified destinations *in a specific order*. The state and action spaces are as in MP.

All maps are 6×6. At every timestep, the agent receives a reward of 1 when achieving the environment objective and 0 otherwise, and the discount factor is $\gamma = 0.99$.

Defining the CMDP Spaces: The environments described above include pairs of state and action spaces. To define a CMDP we couple them with the following context spaces:

Entity Location (EL): The context indicates the locations of core entities in the environment, e.g., passenger locations and drop-off destinations.

Changing Map (CM): The context indicates the number and location of walls in the grid.

Point-of-Interest Order (PO): The context indicates the order of the locations to visit.

Each GN, MP, and PD environment is used with both the EL and CM context spaces. The ON environment is paired with the PO context space. Contexts are represented using CTL representations (see Section 2). For full details on CTL context representations, see Appendix C.

Transfer Session: Each training session begins by randomly sampling two disjoint context sets from the CMDPs described above; the source set C_{src} and the target set C_{tgt}. We adopt "training protocol **B**" of Kirk et al. (2021) such that the size of C_{src} is much smaller than the size of the context space. The agent initially trains on tasks induced by C_{src} for N_{src} steps and then continues its training in C_{tgt} for additional N_{tgt} steps. We record performance progress during and after training. For full details see Appendix E.

Context Representations: We vary the RM information exposed to the agent, using the following representations:

- CTL: controllable context representation without RMs (same baseline in (Toro Icarte et al. 2018)).
- CTL+RS (Toro Icarte et al. 2018): adds dense reshaped RM rewards to the current context's reward functions.
- CTL+LTL+RS (Camacho et al. 2021): adds the *Last Transition Label* (LTL) as an additional context representation that is the current assignment of symbols in $\mathcal{P}$.
- CTL+C-PREP (**ours**): adds the C-PREP context representation: *Desired Transition Label* (DTL), with RS.
- C-PREP (**ours**): C-PREP context representation without a CTL context representation

In Appendix A we show additional experiments using PCG context representations in lieu of CTL.

Reported Metrics: During each training session, we evaluate the source, target, and transferred policies on the context set on which it is trained at 100 uniformly spaced evaluation points. At each evaluation point, we record the policy's average return on 50 sampled contexts. Each training session is repeated 5 times, using different random seeds. From the computed average returns, we calculate the transfer utilities defined in Section 2: JS, TT, and TR (see Appendix F for the formula used to compute these measures). We aggregate the results using interquartile mean (IQM) and calculate the standard deviation and stratified bootstrap 95% confidence intervals (Agarwal et al. 2021). To report results for different performance thresholds, we plot the TT as a function of the threshold. We measure the IQM area under the curve (AUC) of this function, denoted TT_{AUC}.

Results

First, to examine the performance over the entire transfer session, Table 1 shows the interquartile mean (IQM) and standard deviation of the measured transfer utilities (TT_{AUC}, JS, TR) for all tested configurations using a CTL context representation. The best results for each CMDP (row) are marked in bold. Negative TR values that indicate non-beneficial transfer are italicized.

Our method performed best in terms of TT_{AUC} and JS in all but two CMDPs: (1) in GN (shortest horizon) with both context spaces (EL and CM), CTL+LTL+RS performs best in terms of TT_{AUC}; (2) in GN with EL context space, using CTL alone performs best in terms of JS (note that GN with the EL context space has a small context space providing less data for transfer; see Appendix E). Notably, in PD, which is the longest horizon task, our method outperforms all other configurations. Compared to the highest performing baseline, we see TT_{AUC} improvements of 22.84% in the context space CM and 42.31% in the context space EL, and JS improvements of 11.86% in CM and 36.5% in EL. TR results show low and negative TR values for most configurations. Our method is the only one with positive TR throughout all tasks. In the PO environment (longest horizon), we see a performance improvement of over 300% when using C-PREP compared to the next best configuration.

Next, we examine the achieved threshold performance throughout training. Fig. 3 visualizes the IQM TT results, measured in training progress (percentage) as a function of the threshold, i.e., the curve from which we derive TT_{AUC}. The shaded areas are stratified bootstrap 95% confidence intervals. Each row corresponds to a context space. Each column corresponds to an environment. Agents using RS in environments tested with the CM and EL context show similar performance in GN, but a performance gap (in favor of our method) widens as the task horizon grows. In the PO environment, our method is the only one that can be seen converging to a high-performing policy.

Appendix A shows results of experiments using uninformative PCG representations in place of CTL. TT and JS results are similar to those presented with CTL representations. TR results show that for all configurations, it is non-beneficial to use PCG representations for transfer due to severe overfitting. Appendix B presents ablation results that

Utility	Context Space	Environment	CTL	CTL+RS	CTL+LTL+RS	CTL+C-PREP (ours)	C-PREP (ours)
TT_{AUC}	EL	GN	25.41 ± 7.27	6.37 ± 0.71	**6.21 ± 0.42**	6.42 ± 0.44	42.15 ± 3.95
		MP	94.77 ± 6.02	44.78 ± 12.06	19.88 ± 9.36	**18.32 ± 9.27**	86.43 ± 1.89
		PD	97.39 ± 1.57	95.18 ± 3.98	37.58 ± 23.38	**21.68 ± 6.55**	88.35 ± 1.59
	CM	GN	42.18 ± 1.58	16.30 ± 2.35	**7.14 ± 1.41**	7.64 ± 1.72	32.98 ± 0.68
		MP	71.90 ± 10.62	26.48 ± 15.51	14.24 ± 8.85	**13.64 ± 8.60**	47.74 ± 11.46
		PD	86.30 ± 18.81	54.19 ± 21.64	37.52 ± 25.46	**28.95 ± 24.15**	51.02 ± 15.46
	PO	ON	98.04 ± 0.00	95.15 ± 7.66	97.50 ± 5.59	32.93 ± 10.21	**22.54 ± 1.57**
JS	EL	GN	**0.28 ± 0.11**	0.06 ± 0.06	0.07 ± 0.09	0.05 ± 0.05	0.01 ± 0.05
		MP	0.03 ± 0.06	0.26 ± 0.15	**0.48 ± 0.26**	**0.49 ± 0.17**	0.01 ± 0.02
		PD	0.02 ± 0.02	0.03 ± 0.01	0.34 ± 0.20	**0.38 ± 0.12**	0.00 ± 0.01
	CM	GN	0.42 ± 0.03	0.55 ± 0.06	0.73 ± 0.05	**0.75 ± 0.08**	0.49 ± 0.06
		MP	0.23 ± 0.09	0.49 ± 0.14	0.66 ± 0.11	**0.68 ± 0.10**	0.42 ± 0.10
		PD	0.08 ± 0.19	0.28 ± 0.19	0.38 ± 0.27	**0.52 ± 0.25**	0.31 ± 0.19
	PO	ON	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.14 ± 0.26	**0.75 ± 0.02**
TR	EL	GN	*-0.11 ± 0.10*	**0.13 ± 0.04**	0.08 ± 0.03	0.07 ± 0.03	0.04 ± 0.02
		MP	*-0.99 ± 0.01*	**0.24 ± 0.08**	0.24 ± 0.18	0.16 ± 0.12	0.06 ± 0.11
		PD	*-1.00 ± 0.00*	*-0.99 ± 0.21*	*-0.14 ± 0.38*	**0.14 ± 0.09**	*-0.05 ± 0.12*
	CM	GN	*-0.11 ± 0.04*	0.08 ± 0.04	**0.11 ± 0.01**	**0.11 ± 0.01**	0.08 ± 0.04
		MP	*-0.85 ± 0.32*	**0.29 ± 0.13**	0.26 ± 0.06	0.21 ± 0.06	0.07 ± 0.05
		PD	*-0.94 ± 0.25*	*-0.06 ± 0.29*	*-0.05 ± 0.20*	0.06 ± 0.19	**0.07 ± 0.07**
	PO	ON	0.00 ± 0.00	**2.04 ± 6.94**[3]	*-0.87 ± 0.90*	0.69 ± 0.20	0.31 ± 0.03

Table 1: IQM and standard deviation transfer utilities of configurations with CTL. Bold cells indicate best performance in each row. Italic cells indicate negative TR

show that all components of C-PREP are required to achieve the best results. Appendix H shows results for additional experiments on sample efficiency (in terms of number of contexts) and generalization capabilities of C-PREP using PCG.

Discussion

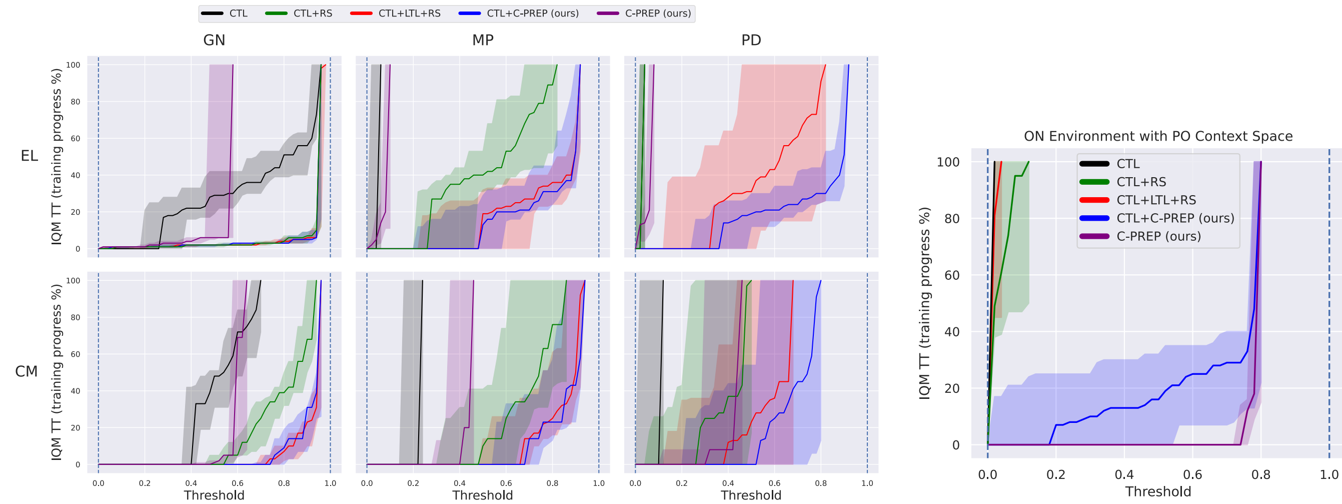
Results demonstrate that C-PREP improves transfer performance in more complex tasks without hindering performance on simpler tasks. As visualized in Fig. 3a, all methods perform similarly in the GN environment (short horizon), but C-PREP opens a performance gap in TT that increases with the difficulty of the environment. The TR results show that only our method is beneficial for transfer in all tasks, as is evidenced by the negative TR values reported for all other configurations. In the PO environment, only agents using C-PREP achieve a threshold greater than 0.2. Furthermore, since the RM in this case differentiates all tasks, it is preferable to use C-PREP without CTL. We observe that the JS performance is approximately 93% of the maximum achieved performance threshold, which is reached in less than 20% of the training progress.

We examine the performance of C-PREP using partial resolution RMs, i.e., some tasks may be represented with the same RM. For this, we use C-PREP alone. In the GN, MP, and PD environments, the agent will achieve a threshold performance of no more than 50% of C-PREP's performance *with* CTL. Fig. 3a shows that C-PREP without CTL achieves medium to low performance depending on the environment and context space. We attribute this to the low coverage of tasks with the partial RM resolution. These RMs (detailed in Appendix D) cover approximately 25% of the tasks in GN and approximately 6% of the MP and PD tasks. We conclude that C-PREP with partial resolution RMs cannot compensate for missing contextual information. However, Fig. 3b shows that in the ON environment, where the RMs are of full resolution, it is preferable to use C-PREP without additional context. We hypothesize that this is due to the large overlap in contextual data between the C-PREP context representation and CTL, making the information in CTL irrelevant. This use of local context illustrates the advantage of solving the context as a series of smaller, simpler contexts.

We show additional results in experiments using PCG in place of CTL in Appendix A to examine the case of uninformative global context representations. Here, we notice that in 60% of the runs, it was more beneficial to train from scratch in C_{tgt} than to transfer from C_{src}. Fig. 4 in the appendix visualizes TT performance of PCG configurations and shows hindered performance compared to those in Fig. 3a that use CTL representations.

Ablation results (found in Appendix B) show the importance of every component of C-PREP. We notice that both TT_{AUC} and JS can be improved by up to 12% in five out of seven tasks by adding the LTL modification to CTL+C-PREP in the most complex task. We hypothesize that this will yield an even greater benefit in scenarios where it is harder to infer the current abstraction label.

Additional results in Appendix H reveal two interesting capabilities of C-PREP. First, C-PREP is more sample efficient in terms of the number of source contexts it is trained on, that is, C-PREP needs to train on fewer source contexts



(a) GN, MP, and PD environments with EL and CM context spaces

(b) ON task with PO context space

Figure 3: The IQM TT of configurations using CTL as a function of the threshold (lower is better).

to achieve similar or better transfer performance than other tested configurations. Second, adding RM information when using PCG significantly improves generalization capabilities at the beginning of training. We see a spike in performance in the first 1M training steps, hinting at the potential of using RMs for learning generalized state representations.

5 Related Work

DRL agents are susceptible to overfitting to the context in which they were trained. Leike et al. (2017) show that small changes to a single detail or obstacle could result in extreme performance degradation. Danesh et al. (2021) demonstrate that simple RL agents overfit to the training settings such that they completely ignore observations. One solution is to train the agent on a distribution of contexts, rather than a single one (Zhang et al. 2018). However, once the context distribution departs from the training distribution, the performance drops despite the knowledge obtained during training (Agarwal et al. 2020). We focus on transferring knowledge to expedite training in novel tasks.

There are several approaches to improve transfer learning in DRL. Meta-learning methods (Finn, Abbeel, and Levine 2017; Wang et al. 2017; Duan et al. 2016) "learn to learn", thereby expediting the agent's adaptation to new surroundings. Model-based methods (Shrestha et al. 2020; Tamar et al. 2016) learn an approximate model of the world, where agents can plan for different contexts using the same model. Bayes-adaptive exploration (Dorfman, Shenfeld, and Tamar 2021; Zintgraf et al. 2019) learn how to best explore new environments. Other methods include disentangling latent representations in various ways to improve adaptation (Dunion et al. 2023a,b). All of the above rely on additional exploration to determine the context before learning to solve it. In contrast, we use contextual information to understand the task a priori to reduce exploration. Using CMDPs, we view the context as additional input to the agent (Langford 2017; Hallak, Di Castro, and Mannor 2015).

To improve few-shot transfer, our method represents contexts as RMs (Toro Icarte et al. 2018). Previous work shows that RMs can be used to expedite learning of a single fully observable or partially observable context (Camacho et al. 2021; Vaezipoor et al. 2021). The latter work uses *linear temporal logic*, which uses structures similar to reward machines with time-related constraints. Note that this is abbreviated LTL, not to be confused with the *last transition label* in this paper. We utilize RMs to represent contextual information, resulting in better sample efficiency and few-shot transfer learning capabilities in multi-context settings.

6 Conclusion

We presented ***Contextual PRE-Planning*** (C-PREP) as a novel context representation function and showed how it enhances zero-shot and few-shot transfer for DRL agents. C-PREP exploits RMs by planning on them and providing the agent with a representation of the next desired transition. Our empirical evaluation demonstrates C-PREP's ability to improve sample efficiency and different transfer utilities, especially for tasks of increasing difficulty.

To focus on improving transfer using RM representations, we assumed the RM generation function is given as input. Future work will include a theoretical analysis of the conditions under which a context representation is guaranteed to enhance transfer and the development of methods for learning to generate appropriate RMs. As a second extension, we intend to examine alternative symbolic representations beyond RMs for enhancing learning and transfer, as well as consider our context representation function's effect in setting in which *options* (Sutton, Precup, and Singh 1999; Illanes et al. 2020) are used to distinguish between sub-tasks. Finally, we plan to examine our representation in real-world settings in which transfer may be beneficial.

References

Agarwal, R.; Machado, M. C.; Castro, P. S.; and Bellemare, M. G. 2020. Contrastive Behavioral Similarity Embeddings for Generalization in Reinforcement Learning. In *International Conference on Learning Representations*.

Agarwal, R.; Schwarzer, M.; Castro, P. S.; Courville, A. C.; and Bellemare, M. 2021. Deep Reinforcement Learning at the Edge of the Statistical Precipice. In *Advances in Neural Information Processing Systems*, volume 34, 29304–29320. Curran Associates, Inc.

Bellman, R. 1957. A Markovian Decision Process. *Indiana University Mathematics Journal*, 6(4): 679–684.

Benjamins, C.; Eimer, T.; Schubert, F.; Mohan, A.; Biedenkapp, A.; Rosenhahn, B.; Hutter, F.; and Lindauer, M. 2022. Contextualize Me – The Case for Context in Reinforcement Learning. ArXiv:2202.04500 [cs].

Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. OpenAI Gym. ArXiv:1606.01540 [cs].

Camacho, A.; Varley, J.; Zeng, A.; Jain, D.; Iscen, A.; and Kalashnikov, D. 2021. Reward Machines for Vision-Based Robotic Manipulation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 14284–14290. Xi'an, China: IEEE. ISBN 978-1-72819-077-8.

Chen, T.; Xu, J.; and Agrawal, P. 2021. A System for General In-Hand Object Re-Orientation. In *Proceedings of the 5th Annual Conference on Robot Learning*.

Danesh, M. H.; and Fern, A. 2022. Out-of-Distribution Dynamics Detection: RL-Relevant Benchmarks and Results. In *Workshop on Uncertainty and Robustness in Deep Learning @ ICML2021*. ArXiv:2107.04982 [cs].

Danesh, M. H.; Koul, A.; Fern, A.; and Khorram, S. 2021. Re-understanding Finite-State Representations of Recurrent Policy Networks. In *Proceedings of the 38th International Conference on Machine Learning*, 2388–2397. PMLR. ISSN: 2640-3498.

Dorfman, R.; Shenfeld, I.; and Tamar, A. 2021. Offline Meta Reinforcement Learning – Identifiability Challenges and Effective Data Collection Strategies. In *Advances in Neural Information Processing Systems*, volume 34, 4607–4618. Curran Associates, Inc.

Duan, Y.; Schulman, J.; Chen, X.; Bartlett, P. L.; Sutskever, I.; and Abbeel, P. 2016. RL^2: Fast Reinforcement Learning via Slow Reinforcement Learning. *arXiv:1611.02779 [cs, stat]*. ArXiv: 1611.02779.

Dunion, M.; McInroe, T.; Luck, K. S.; Hanna, J.; and Albrecht, S. V. 2023a. Conditional Mutual Information for Disentangled Representations in Reinforcement Learning. In *Conference on Neural Information Processing Systems*.

Dunion, M.; McInroe, T.; Luck, K. S.; Hanna, J.; and Albrecht, S. V. 2023b. Temporal Disentanglement of Representations for Improved Generalisation in Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*.

Finn, C.; Abbeel, P.; and Levine, S. 2017. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. *arXiv:1703.03400 [cs]*. ArXiv: 1703.03400.

Gupta, A.; Pacchiano, A.; Zhai, Y.; Kakade, S. M.; and Levine, S. 2022. Unpacking Reward Shaping: Understanding the Benefits of Reward Engineering on Sample Complexity. ArXiv:2210.09579 [cs].

Hallak, A.; Di Castro, D.; and Mannor, S. 2015. Contextual Markov Decision Processes. *arXiv:1502.02259 [cs, stat]*. ArXiv: 1502.02259.

Illanes, L.; Yan, X.; Icarte, R. T.; and McIlraith, S. A. 2020. Symbolic Plans as High-Level Instructions for Reinforcement Learning. *Proceedings of the International Conference on Automated Planning and Scheduling*, 30: 540–550.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In Bengio, Y.; and LeCun, Y., eds., *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Kirk, R.; Zhang, A.; Grefenstette, E.; and Rocktäschel, T. 2021. A Survey of Generalisation in Deep Reinforcement Learning. *arXiv:2111.09794 [cs]*. ArXiv: 2111.09794.

Langford, J. 2017. Contextual reinforcement learning. In *2017 IEEE International Conference on Big Data (Big Data)*, 3–3.

Langford, J. 2018. Real World Reinforcement Learning.

Leike, J.; Martic, M.; Krakovna, V.; Ortega, P. A.; Everitt, T.; Lefrancq, A.; Orseau, L.; and Legg, S. 2017. AI Safety Gridworlds. ArXiv:1711.09883 [cs].

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; Petersen, S.; Beattie, C.; Sadik, A.; Antonoglou, I.; King, H.; Kumaran, D.; Wierstra, D.; Legg, S.; and Hassabis, D. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533. Number: 7540 Publisher: Nature Publishing Group.

Ng, A. Y.; Harada, D.; and Russell, S. 1999. Policy invariance under reward transformations: Theory and application to reward shaping. In *In Proceedings of the Sixteenth International Conference on Machine Learning*, 278–287. Morgan Kaufmann.

Schrittwieser, J.; Antonoglou, I.; Hubert, T.; Simonyan, K.; Sifre, L.; Schmitt, S.; Guez, A.; Lockhart, E.; Hassabis, D.; Graepel, T.; Lillicrap, T.; and Silver, D. 2020. Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature*, 588(7839): 604–609. Number: 7839 Publisher: Nature Publishing Group.

Shrestha, A. K.; Lee, S.; Tadepalli, P.; and Fern, A. 2020. DeepAveragers: Offline Reinforcement Learning By Solving Derived Non-Parametric MDPs. In *International Conference on Learning Representations*.

Sutton, R. S.; Precup, D.; and Singh, S. 1999. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2): 181–211. Publisher: Elsevier.

Tamar, A.; WU, Y.; Thomas, G.; Levine, S.; and Abbeel, P. 2016. Value Iteration Networks. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Taylor, M. E.; and Stone, P. 2009. Transfer Learning for Reinforcement Learning Domains: A Survey. *Journal of Machine Learning Research*, 10(56): 1633–1685.

Terry, J. K.; Black, B.; Grammel, N.; Jayakumar, M.; Hari, A.; Sullivan, R.; Santos, L.; Perez, R.; Horsch, C.; Dieffendahl, C.; Williams, N. L.; Lokesh, Y.; and Ravi, P. 2021. PettingZoo: Gym for Multi-Agent Reinforcement Learning. ArXiv:2009.14471 [cs, stat].

Toro Icarte, R.; Klassen, T.; Valenzano, R.; and McIlraith, S. 2018. Using Reward Machines for High-Level Task Specification and Decomposition in Reinforcement Learning. In *Proceedings of the 35th International Conference on Machine Learning*, 2107–2116. PMLR. ISSN: 2640-3498.

Toro Icarte, R.; Klassen, T. Q.; Valenzano, R.; and McIlraith, S. A. 2022. Reward Machines: Exploiting Reward Function Structure in Reinforcement Learning. *Journal of Artificial Intelligence Research*, 73: 173–208.

Toro Icarte, R.; Waldie, E.; Klassen, T.; Valenzano, R.; Castro, M.; and McIlraith, S. 2019. Learning Reward Machines for Partially Observable Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Torrey, L. A.; and Shavlik, J. 2009. Chapter 11 Transfer Learning. In *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*, 242–264. IGI Global.

Tseng, H.-H.; Luo, Y.; Cui, S.; Chien, J.-T.; Ten Haken, R. K.; and Naqa, I. E. 2017. Deep reinforcement learning for automated radiation adaptation in lung cancer. *Medical Physics*, 44(12): 6690–6705.

Vaezipoor, P.; Li, A. C.; Icarte, R. A. T.; and Mcilraith, S. A. 2021. LTL2Action: Generalizing LTL Instructions for Multi-Task RL. In *Proceedings of the 38th International Conference on Machine Learning*, 10497–10508. PMLR. ISSN: 2640-3498.

Wang, J. X.; Kurth-Nelson, Z.; Tirumala, D.; Soyer, H.; Leibo, J. Z.; Munos, R.; Blundell, C.; Kumaran, D.; and Botvinick, M. 2017. Learning to reinforcement learn. *arXiv:1611.05763 [cs, stat]*. ArXiv: 1611.05763.

Zhang, C.; Vinyals, O.; Munos, R.; and Bengio, S. 2018. A Study on Overfitting in Deep Reinforcement Learning. ArXiv:1804.06893 [cs, stat].

Zintgraf, L.; Shiarlis, K.; Igl, M.; Schulze, S.; Gal, Y.; Hofmann, K.; and Whiteson, S. 2019. VariBAD: A Very Good Method for Bayes-Adaptive Deep RL via Meta-Learning. In *International Conference on Learning Representations*.